Time Series and Anomaly Detection

Gerardo De la O

2022

Real World Example - Anomaly Detection

The first company I worked for specialized in energy management via IoT devices (smart buildings). Every site we managed contained hundreds of devices, and each device collected data values every 10 minutes on average. For simplicity, lets focus on a commercial refrigerator of the corner store variety. One benefit of collecting such data is for anomaly detection, which enables us to diagnose when the refrigerator is close to needing maintenance. Another use case however, is to forecast the energy consumption of the corner store. Reliably estimating trends in kWh usage enables management to allocate the correct amount of funds for utilities and while at the same time implementing practices to save energy.

The following data would be useful to compute a forecast of energy consumption:

1. Refrigerator/Freezer temperature
2. Outside temperature
3. Compressor cycle status (on/off)
4. Door open or close
5. Store hours
6. Visitor count

During store hours, I would expect α to be closer to 0, giving more weight to historical values. This is because there are many unpredictable events occurring during the day that increase randomness, such as people opening/closing the door, the main entry letting warm air in, staff reloading merchandise, etc. As soon as the store closes, however, everything should stabilize, in which case we would favor α closer to 1 and give the sensor more weight to get more accurate readings! In practice, I would toggle α parameters 15 minutes after closing (small to large), and then again 15 minutes before opening (large to small).

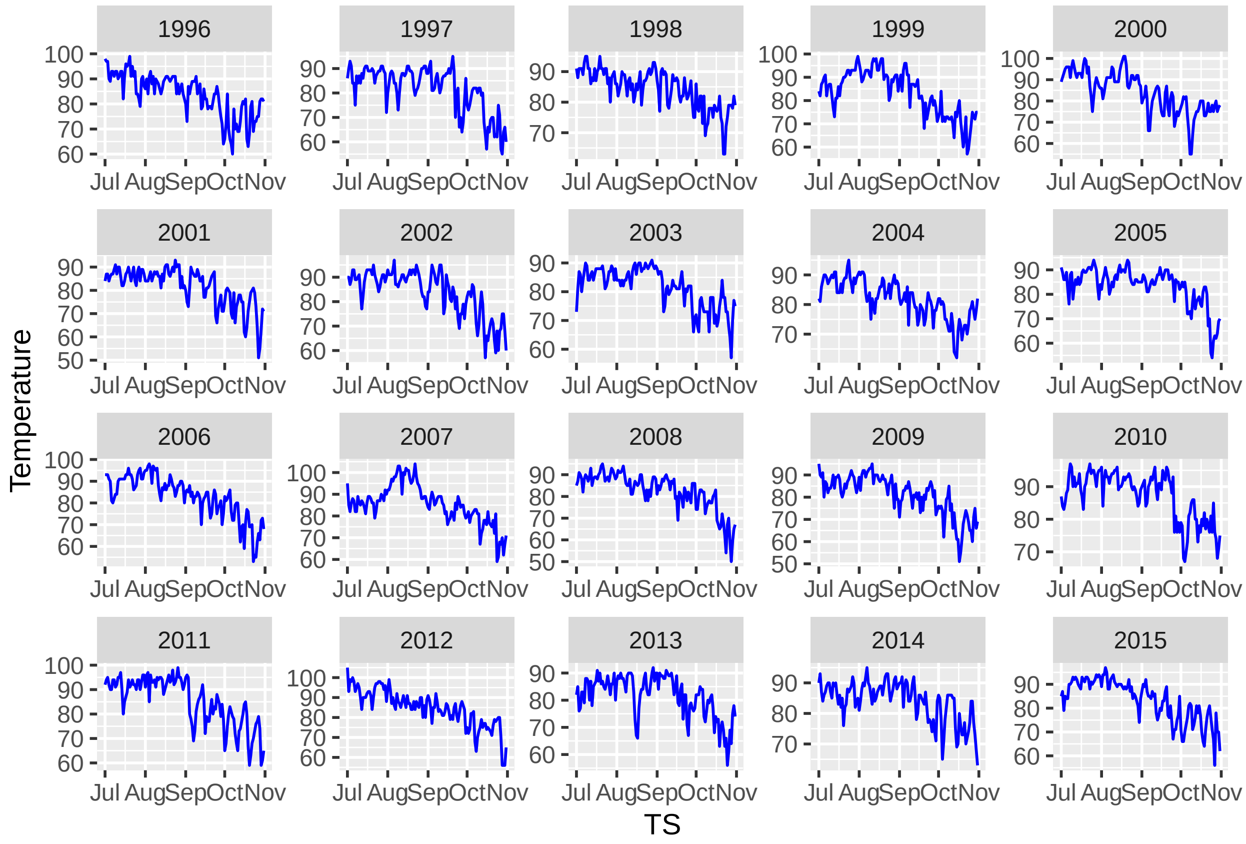
Time-Series Analysis with Exponential Smoothing

This section explores the use of Exponential Smoothing to find out if the unofficial end of summer has shifted in time over the past 20 years. To prepare our data, we first transform it from wide format into time-series (long) format, where each temperature observation is attached to a unique time stamp.

Preparation

```r
library(tidyverse)
library(reshape2)
library(lubridate)

#import data
filename = 'temps.txt'
data <- read.csv(paste0('data 6.2/',filename),sep='\t', check.names=FALSE)
```

```r
#transform into nice, long, juicy time stamp format
data_melted = melt(data, id.vars='DAY', variable.name='Year', value.name='Temperature')
df <- data_melted %>%
    unite(col='TS', c('DAY', 'Year'), sep='-')
df$TS <- as.Date(df$TS, "%d-%b-%Y")

#plot each year
ggplot(data = df, aes(TS)) +
    geom_line(mapping = aes(y = Temperature),color='blue') +
    facet_wrap(~year(TS),scales = "free")
```



Analysis

From the plot above, we see that the data is too noisy for us to easily identify the end of summer. In this section we'll see if we can use exponential smoothing to reduce the noise in our data, then we will run CUSUM algorithm on the smoothed data to see if we can reliably spot a pattern with anomalies.

For Exponential Smoothing, we will uses the es() function from the 'smooth' package. The es() function is very flexible, and has the nice feature of accepting xts objects, which allow us to feed our time stamp column directly! The function generates ETS models, which are a family of time series models that include base, trend (T), seasonal (S), and error elements (E). The code below shows the application of an AAdN model to reduce noise in our data set. Since our goal is to smooth and not predict, our parameters were chosen to favor previous terms (more variance reduction). In the end, we chose $\alpha = 0.2$, $\beta = 0.2$ with no seasonality because the smoothing seemed to fit the data well whilst considerably reducing variance. Finally, we note that ETS models are often used for forecasting, which would likely require a different ETS model and parameters, but that is outside the scope of this project.

```
library(smooth)
library(xts)

head(df)
```

```
##           TS Temperature
## 1 1996-07-01          98
## 2 1996-07-02          97
## 3 1996-07-03          97
## 4 1996-07-04          90
## 5 1996-07-05          89
## 6 1996-07-06          93
```

```
#Generate time object with our actual dates
data_xts = xts(df[,-1], order.by=df[,1])

#Fit model with alpha, beta, and gamma set manually (normally its optimized)

model <- es(data_xts, model='AAdN', persistence=c(0.2, 0.2), h=0)

formula = model$formula
results = model$fitted
model$persistence
```

```
## alpha  beta
##   0.2   0.2
```

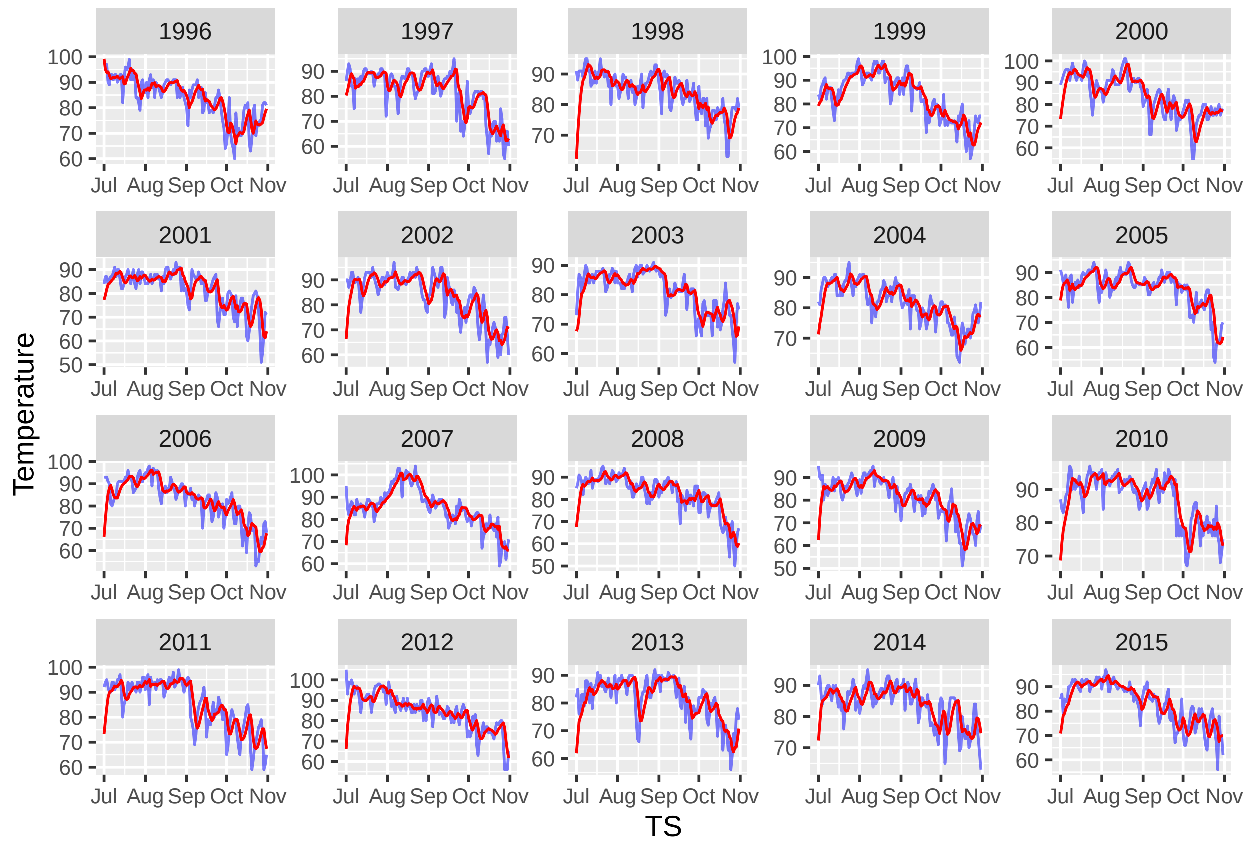
```
model$phi
```

```
## [1] 0.4371784
```

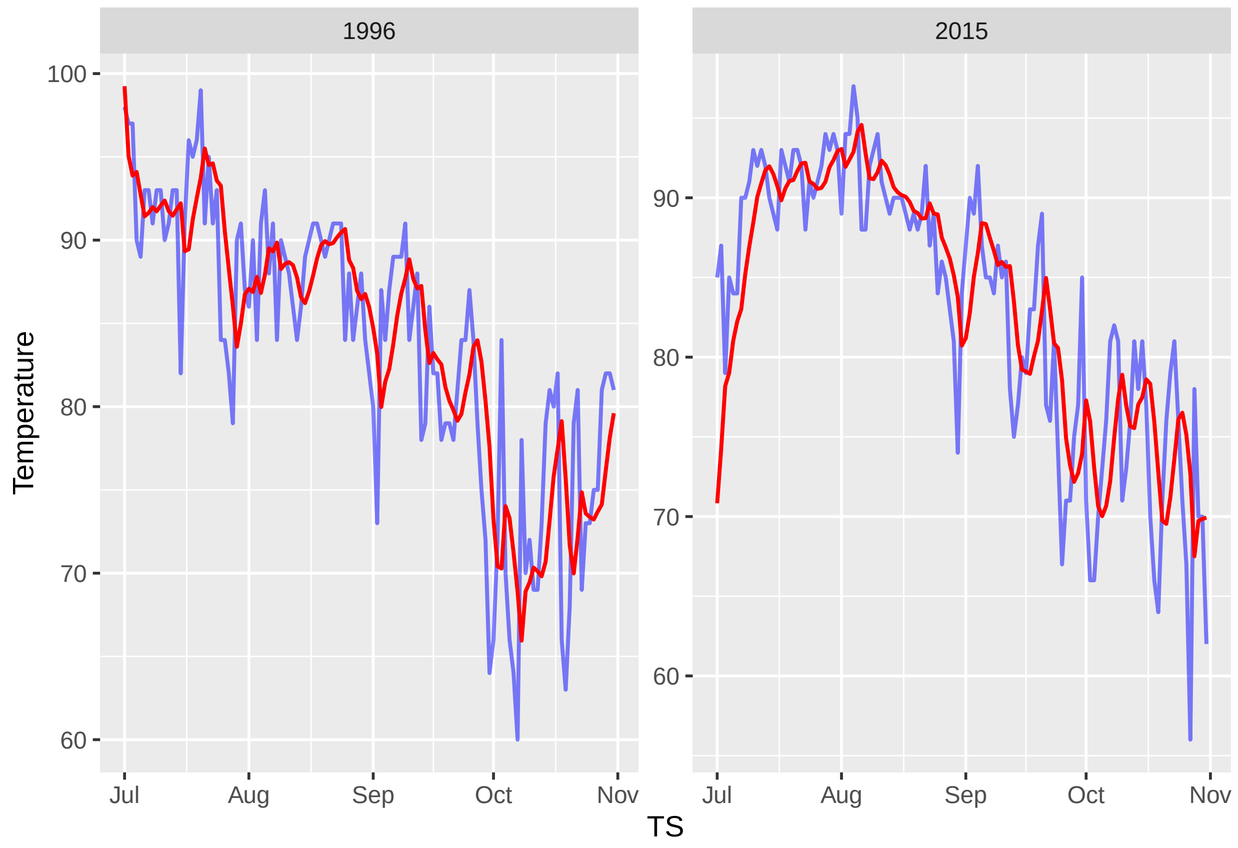
```
# Add smoothed results to our original data
df['ExpSmooth'] = as.numeric(results)
head(df)
```

```
##           TS Temperature ExpSmooth
## 1 1996-07-01          98  99.24342
## 2 1996-07-02          97  95.04185
## 3 1996-07-03          97  93.87658
## 4 1996-07-04          90  94.09372
## 5 1996-07-05          89  92.73887
## 6 1996-07-06          93  91.42981
```

```
# Plot smoothed results over the raw data each year
ggplot(data = df, aes(TS)) +
    geom_line(mapping = aes(y = Temperature),color='blue', alpha=.5) +
    geom_line(mapping = aes(y = ExpSmooth), color='red') +
    facet_wrap(~year(TS),scales = "free") +
    theme( axis.text = element_text( size = 8 ),
           axis.text.x = element_text( size = 8 )
    )
```



```
#Zoom in to get a better look at the smoothing
ggplot(data = subset(df, year(TS) %in% c(1996, 2015)), aes(TS)) +
    geom_line(mapping = aes(y = Temperature), color='blue', size=.7, alpha=.5) +
    geom_line(mapping = aes(y = ExpSmooth), color='red', size = 0.7) +
    facet_wrap(~year(TS),scales = "free")
```

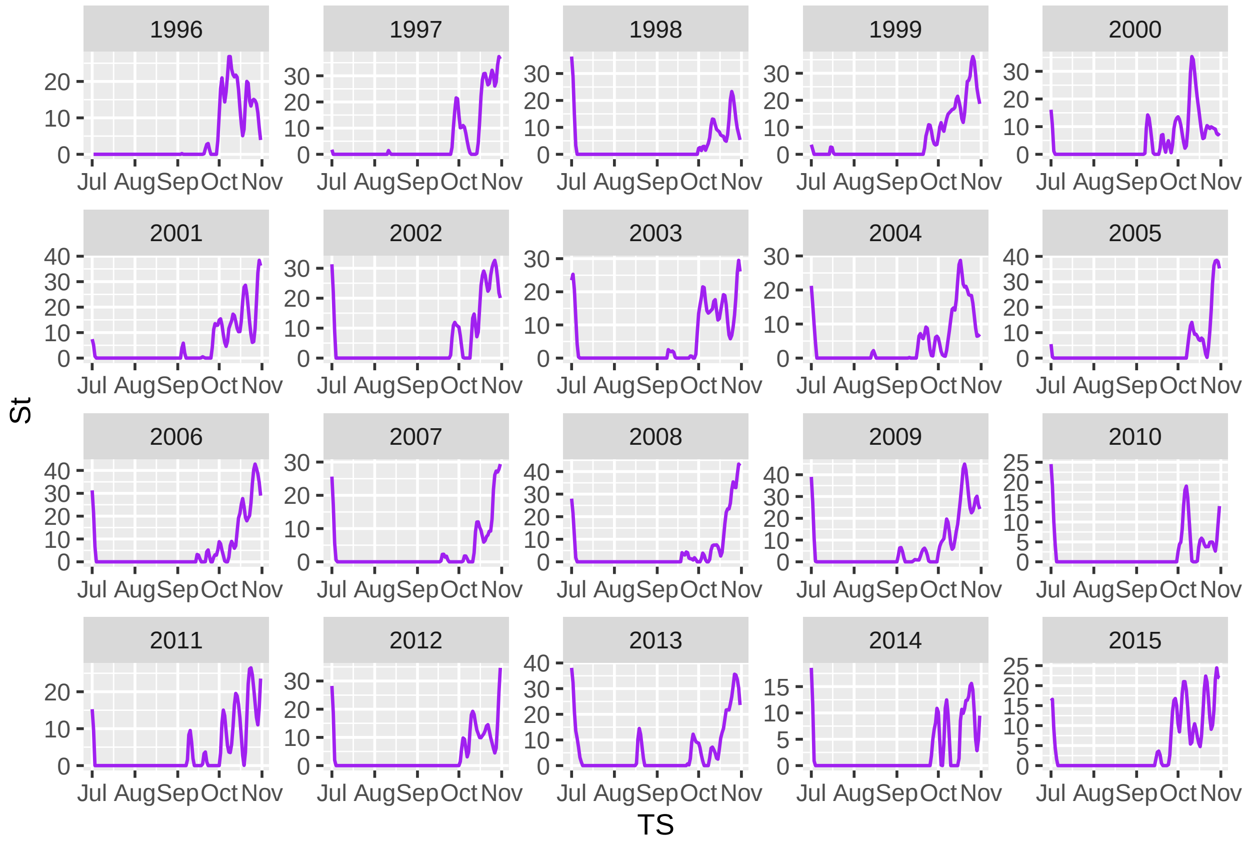


After smoothing, we need a technique to "flag" when anomalous changes in temperature occur. To find these changes, we will use the CUSUM algorithm set to detect decreases. The code below generates a visual that shows where significant drops in temperature occurred, based on our exponentially smoothed data!

```
# Define parameters for CUSUM
mu = mean(df$Temperature)
C = 2.5

# Implement CUSUM
df['mu_X_C'] =  mu - df$ExpSmooth - C

dff <- df %>%
     mutate(S = mu_X_C + lag(mu_X_C)) %>%
     rowwise() %>%
     mutate(St = max(0, S))

print(dff, n=4)
```

```
## # A tibble: 2,460 x 6
## # Rowwise:
##   TS         Temperature ExpSmooth mu_X_C     S    St
##   <date>           <int>     <dbl>  <dbl> <dbl> <dbl>
## 1 1996-07-01          98      99.2  -18.4  NA      NA
## 2 1996-07-02          97      95.0  -14.2 -32.6     0
## 3 1996-07-03          97      93.9  -13.0 -27.2     0
## 4 1996-07-04          90      94.1  -13.3 -26.3     0
## # ... with 2,456 more rows
```

```r
# Plot anomalies every year. Look for a pattern
ggplot(data = dff, aes(TS)) +
    geom_line(mapping = aes(y = St),color='purple', size=.6) +
    facet_wrap(~year(TS), scales = "free")
```



Results

Based on the above analyses, we can observe that the most consistent anomaly across the years tested occurs right around the start of October. There is some variability across years (a few spikes before, a few after), but there is no indication of a consistent pattern that would indicate a systemic shift. Based on these results, and absent more data and more sensitive methods, it is NOT possible to conclude that summer has gotten later in the past 20 years.